



# National Textile University

## Department of Computer Science

Subject:

Operating System

---

Submitted to:

Sir Nasir Mehmood

---

Submitted by:

Eman Babar

---

Reg. number:

23-NTU-CS-FL-1148

---

Semester:

5<sup>th</sup>- A

---

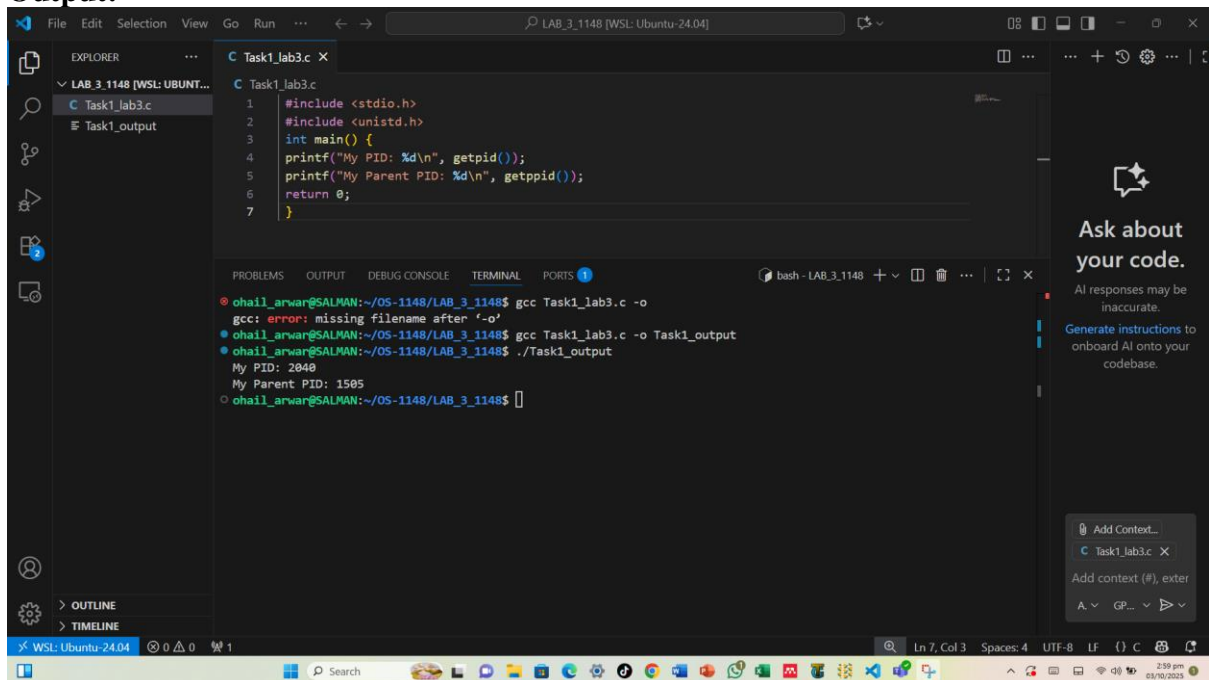
## LAB-03

### Program 1: Print PID and PPID

#### Code:

```
#include <stdio.h>
#include <unistd.h>
int main() {
    printf("My PID: %d\n", getpid());
    printf("My Parent PID: %d\n", getppid());
    return 0;
}
```

#### Output:

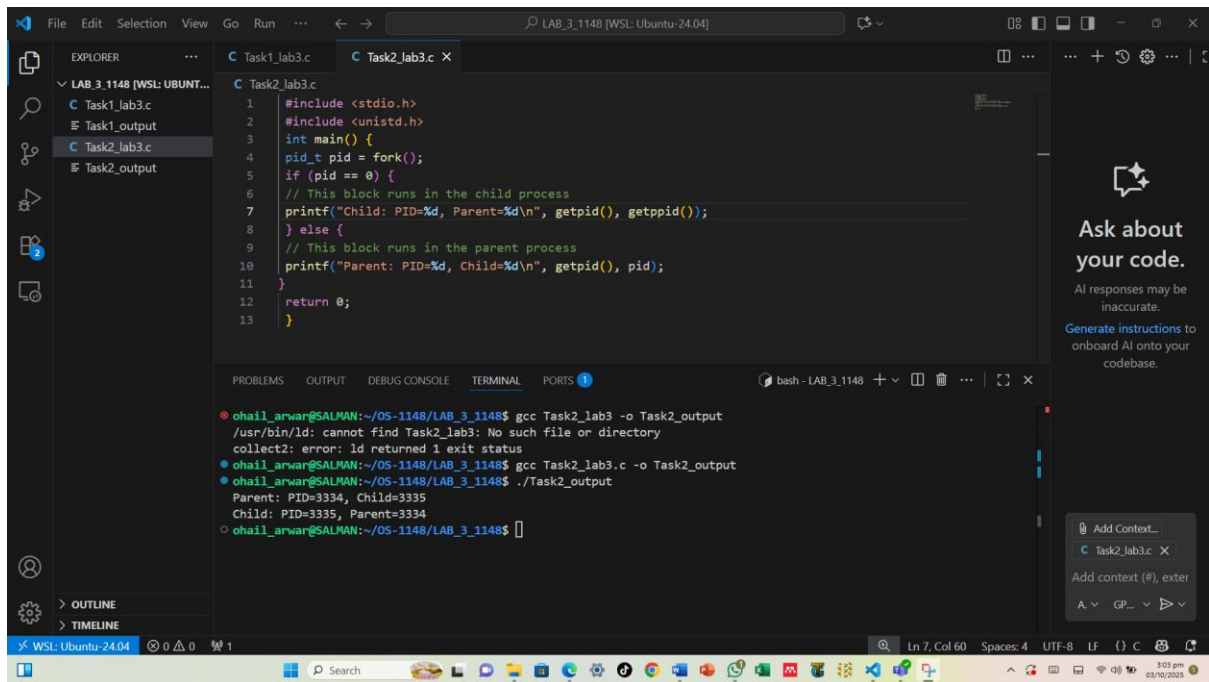
The screenshot shows the Visual Studio Code interface. The Explorer pane on the left shows a project named 'LAB\_3\_1148 [WSL: UBUNTU-24.04]' with files 'Task1\_lab3.c' and 'Task1\_output'. The main editor displays the C code for Program 1. The TERMINAL pane at the bottom shows the execution of the program. It starts with a gcc error: 'gcc: error: missing filename after '-o''. Then, the user runs 'gcc Task1\_lab3.c -o Task1\_output' and './Task1\_output', which produces the output: 'My PID: 2040' and 'My Parent PID: 1505'. On the right side of the terminal, there is an AI assistant sidebar with the text 'Ask about your code.' and 'AI responses may be inaccurate. Generate instructions to onboard AI onto your codebase.'

### Program 2: Fork – Creating Child Process

#### Code:

```
#include <stdio.h>
#include <unistd.h>
int main() {
    pid_t pid = fork();
    if (pid == 0) {
        // This block runs in the child process
        printf("Child: PID=%d, Parent=%d\n", getpid(), getppid());
    } else {
        // This block runs in the parent process
        printf("Parent: PID=%d, Child=%d\n", getpid(), pid);
    }
    return 0;
}
```

#### Output:



### Program 3: Execl – Replacing a Process

#### Code:

```

#include <stdio.h>
#include <unistd.h>
int main() {
    pid_t pid = fork();
    if (pid == 0) {
        execlp("ls", "ls", "-l", NULL);
        printf("This will not print if exec succeeds.\n");
    } else {
        printf("Parent still running...\n");
    }
    return 0;
}

```

#### Output:

The screenshot shows the Visual Studio Code editor interface. The Explorer pane on the left shows a project named 'LAB\_3\_1148 [WSL: UBUNTU-24.04]' with files 'Task1\_lab3.c', 'Task1\_output', 'Task2\_lab3.c', 'Task2\_output', 'Task3\_lab3.c', and 'Task3\_output'. The main editor window displays the code for 'Task3\_lab3.c':

```
1 #include <stdio.h>
2 #include <unistd.h>
3 int main() {
4     pid_t pid = fork();
5     if (pid == 0) {
6         execlp("ls", "ls", "-l", NULL);
7         printf("This will not print if exec succeeds.\n");
8     } else {
9         printf("Parent still running...\n");
10    }
11    return 0;
12 }
```

The TERMINAL pane at the bottom shows the command to compile and run the program:

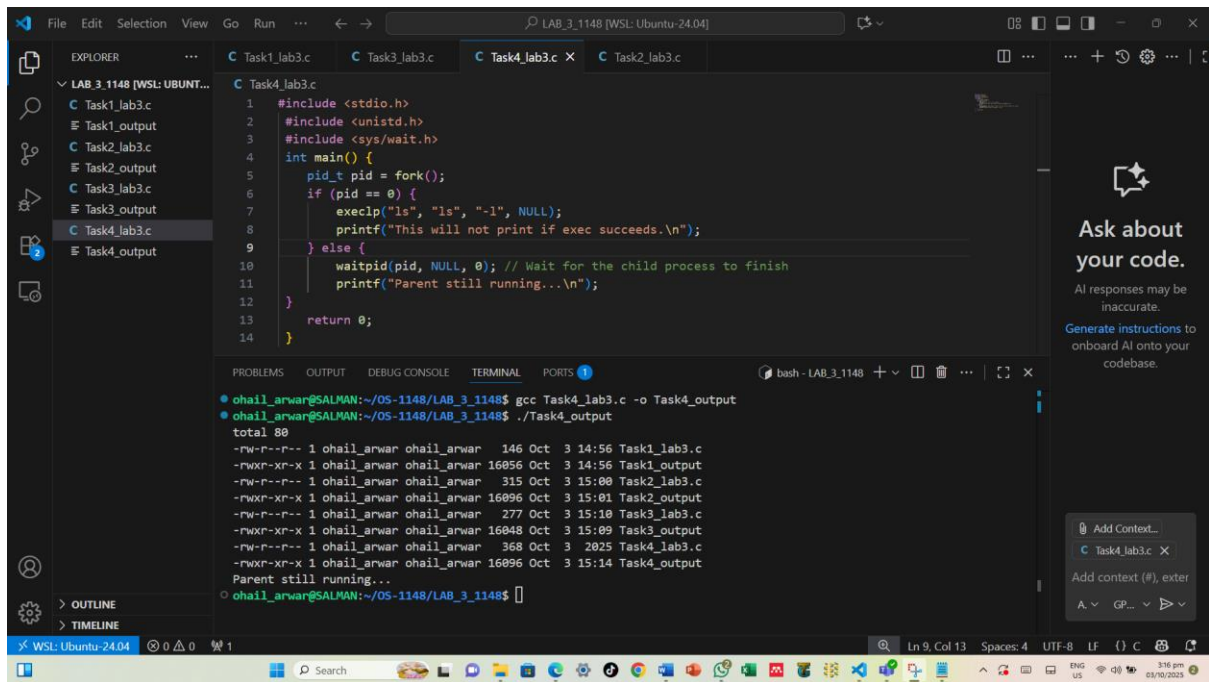
```
ohail_arwar@SALMAN:~/OS-1148/LAB_3_1148$ gcc Task3_lab3.c -o Task3_output
ohail_arwar@SALMAN:~/OS-1148/LAB_3_1148$ ./Task3_output
Parent still running...
total 60
-rw-r--r-- 1 ohail_arwar ohail_arwar 146 Oct 3 14:56 Task1_lab3.c
-rwxr-xr-x 1 ohail_arwar ohail_arwar 16096 Oct 3 14:56 Task1_output
-rw-r--r-- 1 ohail_arwar ohail_arwar 315 Oct 3 15:00 Task2_lab3.c
-rwxr-xr-x 1 ohail_arwar ohail_arwar 16096 Oct 3 15:01 Task2_output
-rw-r--r-- 1 ohail_arwar ohail_arwar 277 Oct 3 15:10 Task3_lab3.c
-rwxr-xr-x 1 ohail_arwar ohail_arwar 16048 Oct 3 15:09 Task3_output
ohail_arwar@SALMAN:~/OS-1148/LAB_3_1148$
```

## Program 4: Wait – Synchronization

### Code:

```
#include <stdio.h>
#include <unistd.h>
#include <sys/wait.h>
int main() {
    pid_t pid = fork();
    if (pid == 0) {
        execlp("ls", "ls", "-l", NULL);
        printf("This will not print if exec succeeds.\n");
    } else {
        waitpid(pid, NULL, 0); // Wait for the child process to finish
        printf("Parent still running...\n");
    }
    return 0;
}
```

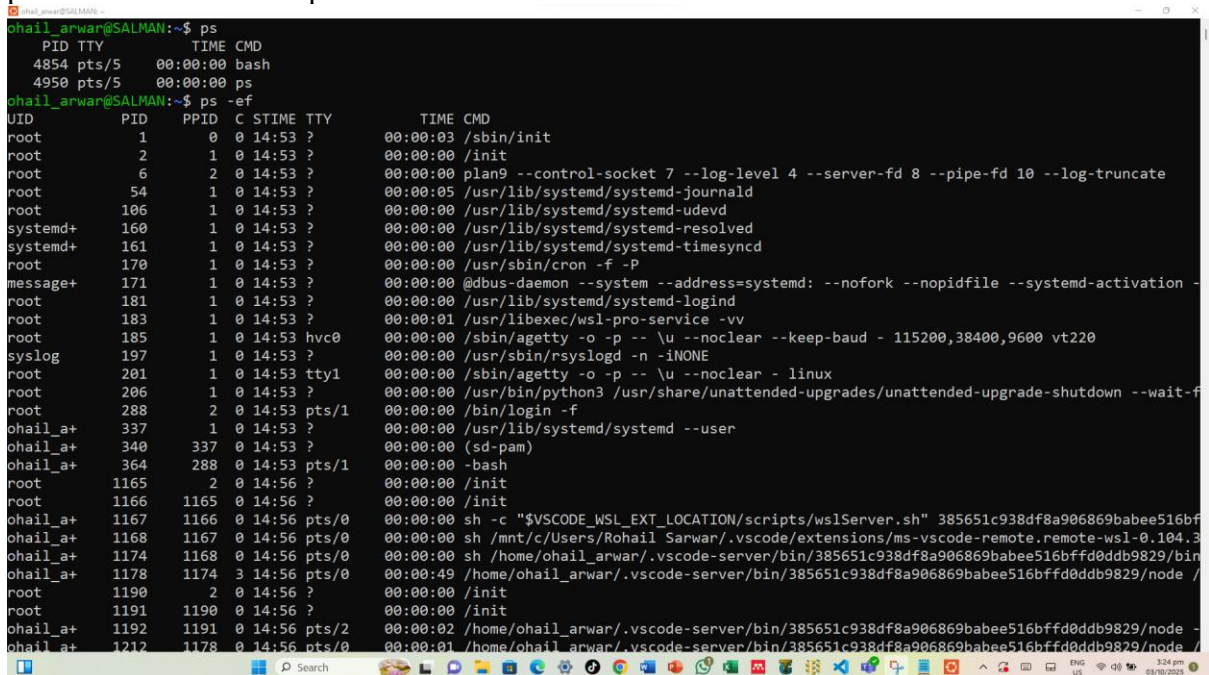
### Output:



## Linux Process Commands

ps → Process Status

ps -ef → Full list of all processes



ps -ef | grep bash → This finds all processes related to the bash shell

top → Dynamic process viewer



```
ohail_arwan@SALMAN:~$ ps -ef
ohail_arwan@SALMAN:~$ ps -ef | grep eman
ohail_arwan@SALMAN:~$ top
top - 15:31:02 up 40 min, 1 user, load average: 0.13, 0.19, 0.18
Tasks: 47 total, 1 running, 46 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.0 us, 2.2 sy, 0.0 ni, 97.8 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 3877.2 total, 2000.5 free, 1823.8 used, 205.9 buff/cache
MiB Swap: 1024.0 total, 1024.0 free, 0.0 used, 2053.4 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM    TIME+  COMMAND
    1 root        20   0   22128   12772   9316 S   0.0   0.3   0:03.85 systemd
    2 root        20   0    3072    1792   1792 S   0.0   0.0   0:00.04 init-systemd(Ub
    6 root        20   0    3104    1924   1920 S   0.0   0.0   0:00.14 init
   54 root       19  -1   66832   15392  14368 S   0.0   0.4   0:06.46 systemd-journal
  106 root       20   0   25000    5888   4736 S   0.0   0.1   0:00.55 systemd-udev
  160 systemd+   20   0   21456   12672  10496 S   0.0   0.3   0:00.19 systemd-resolve
  161 systemd+   20   0   91024    7680   6784 S   0.0   0.2   0:00.31 systemd-timesyn
  170 root       20   0    4236    2432   2304 S   0.0   0.1   0:00.02 cron
  171 message+   20   0    9624    4864   4352 S   0.0   0.1   0:00.53 dbus-daemon
  181 root       20   0   17960    8448   7552 S   0.0   0.2   0:00.36 systemd-logind
  183 root       20   0  1756096  13312  11136 S   0.0   0.3   0:01.07 wsl-pro-service
  185 root       20   0    3160    1920   1792 S   0.0   0.0   0:00.03agetty
  197 syslog     20   0  222508    5248   4352 S   0.0   0.1   0:00.23 rsyslogd
  201 root       20   0    3116    1792   1664 S   0.0   0.0   0:00.01agetty
  206 root       20   0  107032  22528  13312 S   0.0   0.6   0:00.34 unattended-upgr
  288 root       20   0    6664    4480   3840 S   0.0   0.1   0:00.03 login
  337 ohail_arwa  20   0   20132   10880   9088 S   0.0   0.3   0:00.30 systemd
  340 ohail_arwa  20   0   21152    3520   1792 S   0.0   0.1   0:00.00 (sd-pam)
  364 ohail_arwa  20   0    6072    4992   3456 S   0.0   0.1   0:00.06 bash
 1165 root       20   0   30888   1028    896 S   0.0   0.0   0:00.00 SessionLeader
 1166 root       20   0   30888   1036    896 S   0.0   0.0   0:00.02 Relay(1167)
 1167 ohail_arwa  20   0   2800    1664   1664 S   0.0   0.0   0:00.01 sh
 1168 ohail_arwa  20   0   2800    1792   1792 S   0.0   0.0   0:00.00 sh
 1212 ohail_arwa  20   0  1327588  60196  48128 S   0.0   1.5   0:01.12 node
```

## Foreground:

A process that takes control of the terminal until it finishes.  
sleep 30 → You cannot type new commands until it finishes.

## Background:

Add & to run without blocking.  
sleep 30 & → Terminal is free while the command runs.

## Check background jobs:

Jobs

## Bring a job to foreground:

fg %1

## Get PID of a process by name:

pidof sleep

## Run an infinite process:

yes > /dev/null &

## Kill it with:

kill-9 <PID>

## Output:

```
ohail_arwar@SALMAN:~$ jobs
ohail_arwar@SALMAN:~$ sleep 15 &
[1] 5355
ohail_arwar@SALMAN:~$ fg 1
-bash: fg: job has terminated
[1]+  Done                  sleep 15
ohail_arwar@SALMAN:~$ sleep 15 &
[1] 5376
ohail_arwar@SALMAN:~$ fg 1
sleep 15
ohail_arwar@SALMAN:~$ pidof sleep
ohail_arwar@SALMAN:~$ sleep 20 &
[1] 5438
ohail_arwar@SALMAN:~$ kill 5438
ohail_arwar@SALMAN:~$ jobs
[1]+  Terminated          sleep 20
ohail_arwar@SALMAN:~$ yes > /dev/null &
[1] 5519
ohail_arwar@SALMAN:~$ kill 5519
ohail_arwar@SALMAN:~$
```